# DECONV R-CNN FOR SMALL OBJECT DETECTION ON REMOTE SENSING IMAGES

*Wei Zhang, Shihao Wang, Sophanyouly Thachan, Jingzhou Chen, Yuntao Qian*

Zhejiang University
College of Computer Science
Hangzhou, Zhejiang, China, 310058

**ABSTRACT**

Small object detection has drawn increasing interest in computer vision and remote sensing image processing. The Region Proposal Network (RPN) methods (e.g., Faster R-CNN) have obtained promising detection accuracy with several hundred proposals. However, due to the pooling layers in the network structure of the deep model, precise localization of small-size object is still a hard problem. In this paper, we design a network with a deconvolution layer after the last convolution layer of base network for small target detection. We call our model Deconv R-CNN. In the experiment on a remote sensing image dataset, Deconv R-CNN reaches a much higher mean average precision (mAP) than Faster R-CNN.

***Index Terms***— Object detection, Convolutional neural network, Deconvolution, Small object, R-CNN

## 1. INTRODUCTION

With the increasing number of remote sensing images in real-world applications, small object detection becomes more and more important. Among these, the detections of plane and ship in remote sensing images have been a hot topic. Detection of plane and ship in optical images is a "wide range, small object" detection application. The process of remote sensing images has following notable difficulties:

- Object is too small: The objects in remote sensing images are smaller than that in natural images.
- Background is complex: There are too much background objects that are hard to distinguish from remote sensing images.

With the rapid development of convolutional neural networks (CNN), Girshick et al. proposed a framework of R-CNN [1], which converted the object detection problem into classification problem. In this framework, candidate region proposals are extracted by using Selective Search (SS) [2]. Then, the features are extracted from the candidate region proposals by CNN. Finally, R-CNN classifies these features by Support Vector Machine (SVM) classifier, and performs bounding box regression on the candidate region proposals. R-CNN is a pioneering work for object detection with CNN. The results show that CNN are much better than the traditional methods which use hand-engineered features. However, the R-CNN needs to extract more than $2k$ region proposals. Each proposal is fed into the base network (e.g., VGG16) to extract features, and this process is too time-consuming. Besides, R-CNN needs to fix the input size of image in network structure. To improve the efficiency and handle with any input size, Girshick further proposed the Fast R-CNN [3] framework which significantly improved efficiency and the accuracy of R-CNN. Fast R-CNN adopts a region of interest (ROI) pooling strategy, which allows the network extracting high level features on proposal windows with any size much faster. However, Fast R-CNN also uses SS to extract candidate proposals, and the procedure of feature extraction and object classification are separated. Later, Ren et al proposed Faster R-CNN [4], in which the candidate region proposals are obtained by CNN, which is known as Region Proposal Network (RPN). The RPN computes the proposals and shares features with Fast R-CNN. This method can train an end-to-end network and achieve better detection performance. However, after a set of convolution and pooling layers, the feature maps of the last convolution layer in Faster R-CNN are small. The objects in the original image are also much smaller in last feature maps. For example, a $32 \times 32$ object will be $2 \times 2$ when it is passed through a VGG16 [5]. As a result, the object is hard to locate, which makes Faster R-CNN not solve small object detection problem well.

Meanwhile, Fully Convolution Network (FCN) [6] had been proposed and proven to be good at semantic segmentation task. In FCN, a network combined with convolution and pooling, receives an image and outputs the feature maps. Next, the feature maps use deconvolution layers to get an output map with same size as input image. Finally, the input image and output map can be compared to get pixel-based segmentation results. FCN gets a good result in segmentation of PASCAL VOC [7]. Through FCN network, the input image will be down-sampled, and the last feature map is 32 times smaller than the input image. Therefore, it is impossible to do semantic segmentation directly. However, it uses a deconvolution layer to up-sample feature map to the same size as the original image. Finally, it does classification on

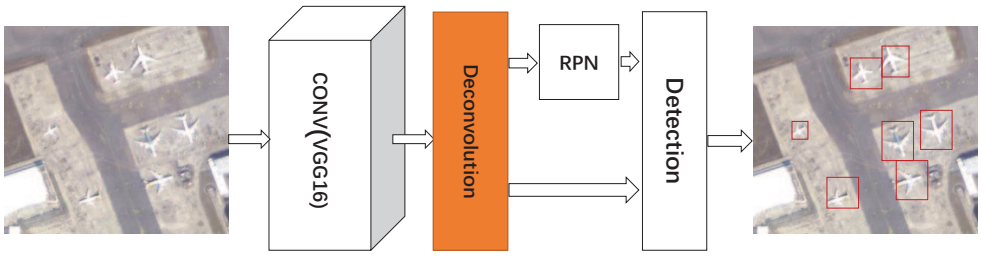



**Fig. 1**. Decon R-CNN network architecture

every pixel of the last feature map to obtain the segmentation result. From the network structure of FCN, we can find that the deconvolution layer will recover information which is lost in the process of extracting features.

Inspired by FCN, in this paper, we propose a Deconv R-CNN, in which we add a deconvolution layer in the base network of Faster R-CNN to recover small object information. The experimental results show that the method achieves good results in the small object detection of remote sensing images. Our main results are: a) On the detection task of ship and plane, we achieve a high mAP [8] of 55.6%, outperforming the Faster R-CNN by 13.1%. b) From the visualization of the detection results, we find that the main improvement comes from the detection of small objects.

## 2. DECONV R-CNN

Figure 1 illustrates the Deconv R-CNN architecture. The image is processed by convolutional layers, which produces the feature maps. Then, we utilize a deconvolution layer to get up-sampled feature maps. Next, a region proposal network (RPN) is used to produce a set of proposals from the up-sampled feature maps. Finally, a Fast R-CNN detection network is used to regress, classify and remove duplication of these proposals to get final detection results.

### 2.1. Deconvolution

Semantic segmentation is understanding an image at pixel level. If we use CNN to semantic segmentation, pooling layers will decrease the resolution, and the information of small object will be loss. Therefore, a good up-sampling method is important. The key contribution of FCN is adopted a deconvolution layers to do up-sampling.

The concept of deconvolution is widely used in image processing and signal processing. Deconvolution is a process used to reverse the effects of convolution on data. With the development of CNNs, deconvolution is also used as a layer for up-sampling in convolution neural network. The deconvolution specific implementation is shown in Fig. 2.

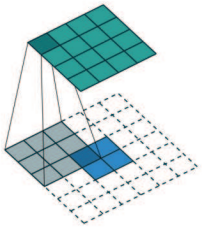


**Fig. 2**. Deconvolution implementation

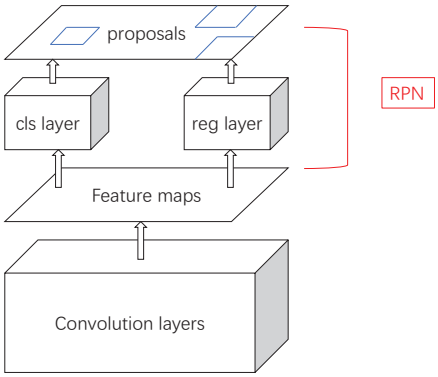



**Fig. 3**. RPN architecture

### 2.2. Region Proposal Networks

A Region Proposal Networks (RPN) receives an image as the input whose size is not specified, and then generates a set of proposals and scores each proposal. The RPN architecture is shown in Fig. 3.

To get these proposals, after the last convolution layer, RPN uses a small network over the feature maps. These feature maps are passed to two convolutional networks, in which one is classification layer (cls) and another is regression layer (reg). Specifically, each pixel in feature maps generates some region candidate boxes. Then, region candidate boxes are fed into the cls and reg layer to get proposals. The cls lay-

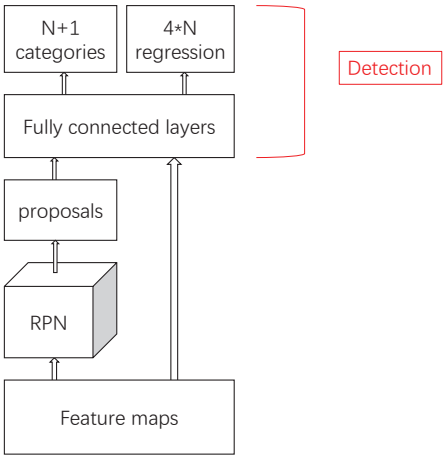



**Fig. 4**. Detection Network

er decides whether these candidate boxes are objects or not, and the reg layer output regresses these candidate boxes to achieve more compact boxes. Finally, the results of cls layer and reg layer are combined as a set of proposals.

### 2.3. Object Detection

The RPN generates a set of proposals. The next step is to determine the label and the position for each proposal. In our model, we use the Fast R-CNN network for detection, which contains 2 fully connection layers and 2 dropout layers. Like RPN, the detection network also has two output layers for each proposal. One is to output $N+1$ label scores (where $N$ is the number of object classes, plus 1 for background) and another is to output $4 \times N$ bounding box regression for each candidate box. The detection network architecture is shown in Fig. 4

### 2.4. Training

To training the RPN, we assign a binary class label (of being an object or not) to each proposal. We assign positive label to a proposal which has an Intersection-over-Union (IoU) higher than 0.7 with any ground-truth box and a negative label to a box if its IoU is lower than 0.3 with any ground-truth box. The multi-task loss function for the RPN is defined as:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum p_i^* L_{reg}(t_i,t_i^*) \quad (1)$$

where $i$ is the index of a proposal and $p_i$ is the probability of a proposal belong to an object. $p_i^* \epsilon \{0,1\}$ is ground-truth label. $t_i^*$ is the ground-truth of the box's position containing 4 values (the coordinates of upper left corner, width and height of the bounding box). $t_i$ is the predicted bounding box. The classification loss $L_{cls}$ is a log loss over two classes (object vs. not object). For the regression loss, we define $L_{reg}(t_i,t_i^*) = R(t_i,t_i^*)$ where $R$ is the robust loss function defined in [3]. The $N_{cls}$, $N_{reg}$ and $\lambda$ are parameters for normalization. In our experiments, $N_{cls}(=256)$ is set to the mini-batch size and $N_{reg}(\approx 2400)$ is the number of region proposal. We set $\lambda = 10$ to balance the two losses.

The stochastic gradient descent (SGD) is used to train our model end-to-end [9]. Firstly, we use a pre-trained model (VGG16) trained on ImageNet [10] for classification to initialize our base network.

All new layers are randomly initialized by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01.

In each iteration, the input images are fed to a set of convolution and pooling layers to extract feature maps. The the feature maps are up-sampled by a deconvolution layer to recover the information of small objects. Next, the feature maps are put into RPN to obtain proposals. Finally, the proposals are fed to the Fast R-CNN detection network.

In the experiment, we train the model for $70k$ iterations where the weight for the momentum in the gradient descent is set to 0.9. The learning rate is set to 0.001 for the first $50k$ iterations and 0.0001 for the remaining 20k iterations with a weight decay of 0.0005.

## 3. EXPERIMENTAL RESULTS

In this section, we evaluate our method and compare it with Faster R-CNN on a remote sensing image dataset. The dataset consists of 2400 images which contains 2 object categories (ship and plane) to detect.

In the first experiment, we randomly sampled 1600 images for training and 800 images for test, and used the mean average precision (mAP) to evaluate the performance. Our method obtains 80.5% mAP, which is slightly better than Faster R-CNN (78.3% mAP).

However, since only a small number of the test images contain small objects, this experiment cannot reflect the detection method's effectiveness in small object detection. To specifically evaluate the performance in small object detection, we selected 40 images that mainly contain small objects as the test set from the 800 test images. We did experiments with different scales of upscaling in the deconvolution layer, which allows us to obtain feature maps with different sizes. The results of 2 times, 4 times and 8 times magnification are shown in Table. 1. We can see that the best results are obtained when up-scaling the feature map by 8 times, which indicates that using the deconvolution layer to upscale feature maps is very helpful in small object detection. The compared results with Faster R-CNN are shown in Table. 2, in which our result is 13.1% higher than that of Faster R-CNN. We al-

**Table 1**. Results of different magnification scales.

| | mAP | Ship | Plane |
|---|---|---|---|
| Deconv-scale×2 | 52.0 | 61.2 | 42.8 |
| Deconv-scale×4 | 54.0 | 61.8 | 46.2 |
| Deconv-scale×8 | **55.6** | **62.4** | **48.7** |

**Table 2**. Comparative results.

| | mAP | Ship | Plane |
|---|---|---|---|
| Faster R-CNN | 42.5 | 50.6 | 34.3 |
| Ours | **55.6** | **62.4** | **48.7** |

so display the detection results on two images obtained by the two methods in Fig. 5 and Fig. 6, respectively. The results show that our method is able to detect small objects with higher accuracy.

Our model, as well as Faster R-CNN, has very high time efficiency. The inference for an image takes about 200ms on a GTX 1080 GPU.

## 4. CONCLUSION

We have presented a method for small object (e.g. ship and plane) detection in remote sensing images. By using a deconvolution layer to recover the information of the small objects lost in the previous pooling layers, our method obtains a great improvement on Faster R-CNN. Moreover, our model can detect the objects in a single image at near real-time frame rate (within milliseconds). The high time-efficiency and the detection precision make our method very useful in real-world application.

## 5. ACKNOWLEDGEMENT

This work was supported by the Nation Natural Science Foundation of China with project No. 61571393.

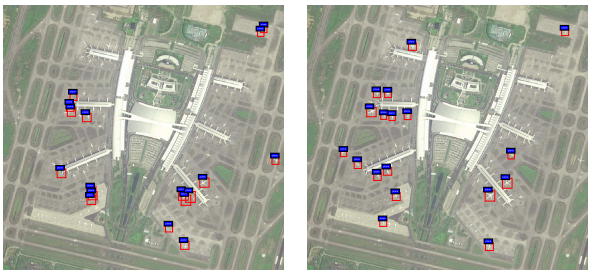
## 6. REFERENCES

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014, pp. 580–587.

[2] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *ICCV*, 2011, pp. 1879–1886.

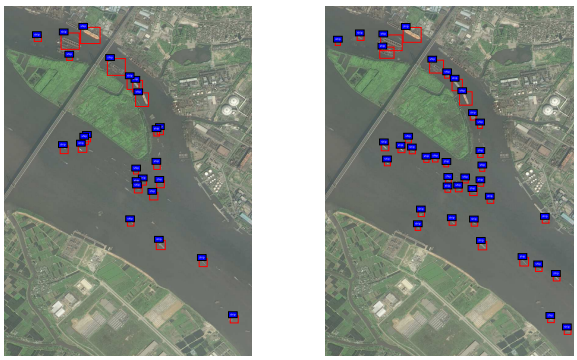
[3] R. Girshick, "Fast r-cnn," in *ICCV*, 2015, pp. 1440–1448.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NIPS*, 2015, pp. 91–99.



(a) Faster R-CNN (b) Ours

**Fig. 5**. Plane detection results.



(a) Faster R-CNN (b) Ours

**Fig. 6**. Ship detection results.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[7] M. Everingham, S. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *Int. J. Comp. Vis.*, vol. 111, no. 1, pp. 98–136, 2015.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comp. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[9] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., "Imagenet large scale visual recognition challenge," *Int. J. Comp. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.